APPENDIX A CLAIMS (STATUS)

1. (Currently amended) A computer system ~~providing~~ comprising an object-based virtual machine environment using multiple heaps enabling a single virtual machine to be resettable thereby avoiding a need to terminate and start a new virtual machine, in which middleware runs successive applications on a single virtual machine, said system including storage for storing objects for running said applications, said storage being logically divided into three heaps:

a system heap wherein system classes for said single virtual machine are loaded, linked, verified initialized and compiled into ~~the~~ said system heap, thereby providing subsequent applications with the ability to use these classes without having to reload them and which said system heap is not garbage collected, wherein subsequent applications reuse said classes in said system heap thus eliminating the need for repeating the reloading, linking verifying and compiling of said classes;

a middleware heap which is garbage collected between successive applications; and wherein reusable objects other than class objects that must persist between successive applications, are stored in said middleware heap ~~and~~

a transient heap which is entirely cleared in between successive applications, and which has no garbage collected within the lifetime of an application, said transient heap, being used for storing applications of objects that are used for only the duration of an application, any objects that are pointed to by objects in said system are moved into said middleware heap and including a middleware classloader and an application class loader, wherein objects from classes loaded by the middleware classloader will be created in said middleware heap, and objects from classes loaded by the application classloader will be created in said transient heap,

said system operating in the absence of a proxy area and a stack map;

the use of said three heaps enabling a single virtual machine to retain data and objects across multiple applications

the use of said transient heap providing a more efficient method of garbage collection that

enables said virtual machine to quickly reset the transient heap thereby creating a a greater virtual machine.

2. (Canceled) The computer system of claim 1, wherein system classes for the virtual machine are loaded into the system heap, thereby providing subsequent applications with the ability to use these classes without having to reload them

3. (Canceled) The computer system of claim 2, wherein reusable objects other than class objects that must persist between successive applications are stored in the middleware heap.

4. (Canceled) The computer system of claim 1, wherein the middleware heap is garbage collected between successive applications.

5. (Canceled) The computer system of claim 1, wherein only the portion of storage corresponding to the middleware heap is garbage collected between successive applications.

6. (Canceled) The computer system of claim 1, wherein the transient heap is used for storing applications objects that are used for only the duration of the application.

7. (Previously Presented) The computer system of claim 1, wherein at the end of an application, any objects in the transient heap that are eligible for use by the next application, and which are referenced by live objects in the system heap or middleware heap, are promoted to the middleware heap.

8. (Previously Presented) The computer system of claim 1, further comprising a card table, in which each card corresponds to a portion of the middleware heap, and said card is marked if the middleware heap potentially references an object in the transient heap.

9. (Previously presented) The computer system of claim 8, wherein each card corresponds to a memory region having a size greater than the minimum size for an object.

10. (Previously presented) The computer system of claim 9, wherein a card is marked whenever an object in the corresponding memory region is updated.

11. (Canceled) The computer system of claim 1, further including a middleware classloader and an application class loader, wherein objects from classes loaded by the middleware classloader will be created in the middleware heap, and objects from classes loaded by the application classloader will be created in the transient heap.

12. (Previously presented) The computer system of claim 11, further including one or more system classloaders, wherein objects from classes loaded by the one or more system classloaders are created in the middleware heap or the transient heap depending on the current context.

13. (Previously presented) The computer system of claim 12, wherein the current context is middleware if the method being run derives from a class loaded by the middleware classloader, and application if the method being run derives from a class loaded by the application classloader.

14. (Previously presented) The computer system of claim 13, wherein if the method being run derives from a class loaded by said one or more system classloaders, then the current context retains the value it had immediately before the method was run.

15. (Currently amended) A method of operating a computer system providing an object-based virtual machine environment, using multiple heaps enabling a single virtual machine to be resettable thereby avoiding a need to terminate and start a new virtual machine, in which middleware runs successive applications on a single virtual machine, said system including storage for storing objects for running said applications, said method comprising the steps of:

logically dividing the storage into three heaps:

a system heap wherein system classes for said single virtual machine are loaded, linked, verified initialized and compiled into ~~the~~ said system heap, thereby providing subsequent applications with the ability to use these classes without having to reload them,

a middleware heap, which is garbage collected between successive applications, and wherein reusable objects other than class objects that must persist between successive applications, are stored in said middleware heap;

and a transient heap;

performing garbage collection on the said middleware heap between successive

applications; and wherein reusable objects other than class objects that must persist between successive applications, are stored in said middleware heap and performing garbage collection on said the transient heap, but not on the said system heap; and

clearing the transient heap in between successive applications, said transient heap being used for storing applications objects that are used for only the duration of an application and which has no garbage collected within the lifetime of an application.

16. (Currently amended) A computer program product comprising computer program instructions encoded on a computer readable media for loading into a computer system which provides an object-based virtual machine environment, using multiple heaps enabling a single virtual machine to be resettable thereby avoiding a need to terminate and start a new virtual machine, in which middleware runs successive applications on a single virtual machine, said system including storage for storing objects for running said applications, said instructions causing the computer system to perform a method comprising the steps of:

logically dividing the storage into three heaps:

a system heap wherein system classes for said single virtual machine are loaded, linked, verified initialized and compiled into ~~the~~ said system heap, thereby providing subsequent applications with the ability to use these classes without having to reload them,

a middleware heap, which is garbage collected between successive applications; and wherein reusable objects other than class objects that must persist between successive applications, are stored in said middleware heap,

and a transient heap;

performing garbage collection on the said middleware heap between successive applications; and wherein reusable objects other than class objects that must persist between successive applications, are stored in said middleware heap and performing garbage collection on said the transient heap, but not on the said system heap; and

clearing the transient heap inbetween successive applications, said transient heap being used for storing applications objects that are used for only the duration of an application and which has no garbage collected within the lifetime of an application.